REMARKS

Claims 1, 4-21 and 31-40 are pending. Claims 1, 20, 21 and 31 are amended to improve the clarity of the claim language. Claims 32-40 are withdrawn.

Objections to the Claims

Claim 21 is objected to because of an informality. Applicants amend claim 21 for the purpose to improve the clarity of the claim language as suggested by the examiner. Accordingly, Applicants respectfully request withdrawal of this objection.

35 U.S.C. § 103 Rejection

Claims 1, 4-21 and 31 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Scheifler et al. (US Patent 6,138,238) in view of Colburn et al. (US Patent 6,173,404). Applicants respectfully traverse this rejection.

Applicants respectfully submit that Scheifler and Colburn are being read too broadly. Scheifler and Colburn's security paradigm is based on permissions being granted based on source, executor, or owner of objects. While some of their security features are incorporated into objects, the structure and functionality of Scheifler and Colburn, alone or in combination, are different than the claimed invention.

Combining Colburn with Scheifler impermissibly changes the principle operation of Scheifler. Scheifler stores security details (e.g., permissions) in a centralized policy file not in target objects as claimed. See e.g., Figure 4. Permissions authorizing access are based on <u>source</u> and <u>executor</u> of a piece of particular code. Scheifler explicitly states

> A security enforcement mechanism is provided in which the access permissions of a thread are allowed to vary over times based on the source and executor of the code currently being executed. The source of the code indicates whether the code is from a trusted or untrusted source. The executor indicates the principal on whose behalf the code is being executed. For example, the executor may be a particular user or a particular organization on whose behalf the process or program is operating on a client computer. . . . According to an implementation consistent with the present invention, the security mechanism described herein uses permission objects and protection domain objects to store information that models the security policy of a system. The nature and use of these objects, as well as the techniques for dynamically determining the time-variant access

> privileges of a thread, are described hereafter in great detail. Col. 7, line 65 – col. 8, line 30.

Colburn relies on an owner-identifier being incorporated into objects. This identifier is based on the creator of an object or the system used to create the object. So that Colburn's system may function, Colburn defines a set of access authorizations that creators must implement into their objects. Colburn's system is not based on a centralized authority controlling security details, but the existence of an owner identifier and a standardized system of access authorizations. See e.g., Abstract, col. 8, line 60 – col. 9, line 41, and col. 10, lines 6 – 51. Combination of Colburn with Scheifler requires abandoning Scheifler's use of a centralized authority to determine security. Conversely, incorporating Scheifler into Colburn requires Colburn to adopt Scheifler's use of centralized permission objects. Both systems describe two different specific implementations of controlling access that are not compatible. For this reason alone, the combination is improper and the *prima facie* case of obviousness has not been met.

However, even a combination does not teach or suggest the claimed invention. The examiner states

> Scheifler does not expressly teach the method, system and computer readable medium comprising: wherein the call from an object; the target object determining access to the other interfaces; and wherein the determination step comprising means for examining a security policy contained entirely within the target object....by combination Colburn's target security scheme with Scheifler's permission implementation, the resulting combination further teaches the target object implementing access authorization in association with implied permission to other interfaces, as the target object determines the access authorization of the received call to the other interfaces by examining the target object's own security policies. It would have been obvious for one of ordinary skill in this art, at the time of invention was made to include Colburn's inter-object security scheme into Scheifler's object for the benefit of implementing a more robust security scheme between objects (Colburn, col. 3, ll. 34-37) to obtain the invention as specified in claims 1, 20-21 and 31. Office Action, p. 4-5.

Scheifler's disclosure of implied permissions to other interfaces does not read upon the present claims. Therefore, implied permissions in the combination of Scheifler and Colburn cannot read upon the present claims regardless of the implementation. The present claims do recite that permissions are implied as described in Scheifler. As stated, interface permissions in the present invention as claimed are not implied. Each interface may grant varying degrees of access to the

target object. See e.g., Specification para. [0058]. Scheifler's disclosure of implied permission does not constitute determining access to other interfaces of a target object as the examiner implies. Scheifler explicitly states:

> If a permission is represented by a permission object, the validation method for the permission object contains code for determining whether one permission is implied by another. For example, a permission to write to any file in a directory implies a permission to write to any specific file in that directory, and a permission to read from any file in a directory implies a permission to read from any specific file in that directory. However, a permission to write does not imply a permission to read. Col. 12, lines 46-55.

In the present invention, access to one interface does not "imply" access to another interface. See, e.g., Specification, para. [0056] ("Each of these interfaces grants a specific set of permissions to any object obtaining a reference to it…"). As Scheifler explicitly states, the permission object contains code for determining whether one permission is implied by another. The present invention as claimed does not determine whether a permission is implied based on another permission. Rather the target object determines whether an external object access to a particular interface based on a call to the first interface. See e.g., Specification, para. [0058].

Adding Colburn to Scheifler does not solve either's deficiencies. Colburn's inter-object security scheme requires an owner identifier incorporated into the objects:

> The owner identifier includes identification of the user, person, or entity (e.g., corporation) who or that creates the object, or identification of a computer system used by the user, person, or entity to create the object definition. The owner identifier provides a basis for distinguishing the creator of an object from the user of that object. This distinction allows instances of objects created by others to be given fewer access permissions or rights than the user implementing the object. Col. 1, lines 55 – 63.

To resolve the security status of an owner identifier, Colburn requires identification of the entity that creates an object definition and access is granted with regard to the computer system. Security permissions are not granted based on a call to a first interface and the security policy of a target object is certainly not contained solely within the target object as claimed. Colburn states

> Exemplary owner identifiers include identification of the user, person, or entity (e.g., corporation) who or that creates the object definition, or identification of a computer system used by the user, person, or entity to create the object definition. The object identifier may be in the form, for example, of a pointer to the owner

> IThing COM object. For purposes of illustration, process 180 is described as if the Owner identifier references the individual user who creates the object definition. It will be appreciated, however, that the Owner identifier could alternatively identify an entity (e.g., corporation or educational institution) where or a computer system on which the object definition is created.
>
> Process block 190 indicates that the object is made available for use, either by the user who created the object or one or more other users.
>
> Process block 192 indicates that a selected user instantiates on the selected user's computer system an accessing instance of an object (e.g., accessing instance 156) based upon the object (e.g., object 154).
>
> The accessing object will seek access to a target (e.g., a method or property) of a target instance (e.g., target instance 164) on the selected user's computer system.
>
> The selected user has a set of permissions or rights with regard to the computer system that allow the user to access a wider range of computer system resources than can be accessed by other users, particularly users who are unknown to the system. In addition, different classes of target object services have different access authorizations (e.g., access authorizations 194, FIG. 8) that represent different permissions or rights for performing different access operations. The target access authorizations indicate which objects are allowed access to particular services on a target object.
>
> In one implementation, three levels of access authorizations 194 are All, Owner, and Exemplar. Object services or targets with the access authorization All indicate that all objects, regardless or their Owner, may perform access operations on the targets. Targets with the access authorization Owner indicate that only the owner of the target instance may access it. Object services with the access authorization Exemplar indicate that only services defined on the same exemplar as the target instance may access those targets. In this implementation, the Exemplar access authorization is the most restrictive and the All access authorization is the least restrictive. It will be appreciated, however, that other access authorization levels could be utilized, relating to a variety of considerations. Col. 10, line 60 – col. 11, line 38.

Access is granted based on a combination of owner identifier and access authorizations. The access authorizations are not interface based but arbitrary designations that enable different levels of access to the objects. The access authorizations are dependent on whether a particular object belongs to an object security class. See e.g., Colburn, claim 1 ("An access authorization security condition associated with the service and conditioning access to the service by the second object on whether the second object is in one of plural object security classes, one of

which being a class in which the first and second objects are defined by the same person or entity."). In order for the access to be resolved the owner identifier must be resolved, which necessarily involves a process outside of a particular object. For example, Colburn states

> Process block 260 indicates that the server restricts the user (and the client computer) to targets having access authorizations of All.....Process block 262 indicates that accessing objects from the user (and the client computer) are permitted to access targets on the server having Owner access authorization and in which the user is the owner. This access is permitted through targets on a call stack on the server. Col. 14, lines 5 – 13.

Moreover, Colburn's discussion of dynamic inheritance is further evidence that security is not determined as claimed. See e.g., Fig. 11 and corresponding description col. 14, line 35 – col.16, line 21.

The present invention as claimed supported by the specification recites that

> The decisions that each object makes as to how, when, and with whom to share its own interfaces or those of another object are up to the specific object implementation, and can be dependent on any number of factors that the object chooses to consider. In the preferred embodiment of the present invention, the primary or sole factor used to determine which interfaces of a callee can be successfully requested by a caller is simply which interface the request was made using. While other factors such as cryptographic signatures, password protection, and user verification may be used by any particular object to establish special rules based on particular situational needs, using these types of factor would require a central authority to be contacted and for certain tests to be performed on the caller to determine whether or not the caller has the necessary permissions. If an objects gets a call on one of its interfaces, the objects knows that the calling object has authority or permission to make that call, by virtue of having had a reference to that interface in the first place. In a preferred embodiment within any given object, the knowledge of which particular interface was used to request another is all of the information that is needed to determine which interfaces get exposed in response to that request, and thus how the trust network is built. Thus, constant permission checks with a central authority for each call to an object interface are not necessary.
>
> Security policies can be established by dividing an object's functionality into interfaces that define categories of services grouped by what level or type of permission is required to access the service. For example, an object can have an interface that grants minimal access and can give this capability to objects that cannot necessarily be fully trusted, and can provide other interface that grant full access to all capabilities to other objects that can be trusted. Paras. [0052] – [0053].

Scheifler's security model requires a centralized authority to determine permissions, which does not read on the claims. Incorporation of Colburn impermissibly requires modifying the principle of operation of Scheifler, or in the alternative, the combination still requires a centralize authority to determine permissions. In either case, neither reference discloses permissions based on the call to the first interface as claimed. Scheifler and Colburn, individually and in combination, do not teach or suggest each and every element of the claims. Accordingly, the Applicants respectfully request withdrawal of this rejection.

Conclusion

All of the stated grounds of rejection have been properly addressed. Applicants therefore respectfully request that the examiner reconsider the outstanding rejections and allow the present claims. The examiner is invited to telephone the undersigned representative if an interview might expedite allowance of this application.

Respectfully submitted,
BERRY& ASSOCIATES P.C.

Dated: November 24, 2010

By: /Shawn Diedtrich/
Shawn Diedtrich
Registration No. 58,176
Direct: 480.704.4615

9229 Sunset Blvd., Suite 630
Los Angeles, CA 90069
(310) 247-2860